

**Internship Report**

Machine Learning for Recommendations

and Profile Modeling in Social Networks

Student: ZHOU Juncheng

Internship supervisors: Prof. Albert Bifet, Prof. Themis Palpanas

Internship of Master Data & Knowledge in 2018

April 1st, 2018 – September 28, 2018

**Acknowledgement**

Before the development on this professional experience, I would like to thank all people who contributed to the success of my internship and even those who helped me to make this internship a very profitable moment.

I would like to thank my internship supervisor, Prof. Themis Palapanas, who helped me and accompanied me throughout this professional experience with great patience and pedagogy. Also, I would like to thank my professor Prof. Albert Bifet, who lead me into Machine learning and accepted to be my internship supervisor.

Finally, I would like to thank all the employees of company, and they gave me some useful advice during the internship.

**Summary**

Contents

[2 Introduction 5](#_Toc525916818)

[3 Work Schedule 5](#_Toc525916819)

[4 Presentation of Laboratory & Lounjee 6](#_Toc525916820)

[5 Presentation of Project 7](#_Toc525916821)

[5.1 Existing context 7](#_Toc525916822)

[5.2 Presentation of project 8](#_Toc525916823)

[5.2.1 Performance of Lounjee’s algorithm **Error! Bookmark not defined.**](#_Toc525916824)

[6 Project Implement: Data preprocessing 9](#_Toc525916825)

[6.1 Data Acquisition 9](#_Toc525916826)

[6.2 Data Engineering 10](#_Toc525916827)

[6.2.1 Data Extraction 10](#_Toc525916828)

[6.2.2 Data preprocessing 10](#_Toc525916829)

[7 Data Analysis 13](#_Toc525916830)

[7.1 Data integrity analysis 13](#_Toc525916831)

[7.2 Data balance analysis 13](#_Toc525916832)

[7.2.1 Classes Weight 14](#_Toc525916833)

[7.2.2 Sampling 15](#_Toc525916834)

[8 Data engineering 15](#_Toc525916835)

[8.1 One hot encoding 15](#_Toc525916836)

[8.2 Random under sampling & random over sampling 16](#_Toc525916837)

[8.3 Merge datasets 17](#_Toc525916838)

[9 Model Introduction 18](#_Toc525916839)

[9.1 Popular Model: Collaborative filtering 18](#_Toc525916840)

[9.2 Traditional model: Matrix factorization model 19](#_Toc525916841)

[9.3 Advanced model: Deep learning model 20](#_Toc525916842)

[10 Model Building 20](#_Toc525916843)

[10.1 Backend and framework 20](#_Toc525916844)

[10.2 Deep learning model architecture 21](#_Toc525916845)

[10.2.1 Embedding layer 21](#_Toc525916846)

[10.2.2 Dropout layer 22](#_Toc525916847)

[10.2.3 Activation: LeakReLU 23](#_Toc525916848)

[10.2.4 Last Activation: Softmax 24](#_Toc525916849)

[10.2.5 Connected layer: Dense 25](#_Toc525916850)

[10.2.6 Model Compile 27](#_Toc525916851)

[11 Model training 27](#_Toc525916852)

[11.1 Callback Function 27](#_Toc525916853)

[11.2 Batch size 27](#_Toc525916854)

[11.3 Model Performance Testing 28](#_Toc525916855)

[11.4 Performance of Model **Error! Bookmark not defined.**](#_Toc525916856)

[12 Prediction 29](#_Toc525916857)

[13 Result Sort 29](#_Toc525916858)

[14 model save & model load 30](#_Toc525916859)

[15 Restful service for Lounjee 31](#_Toc525916860)

[16 The versions what we did of recommendation system 32](#_Toc525916861)

[16.1 version 1: Deep Learning Regression 32](#_Toc525916862)

[16.2 Version 2: Deep Learning Classification 32](#_Toc525916863)

[16.3 Version 3: XGBClassifier （Split data） 33](#_Toc525916864)

[17 New Data coming 34](#_Toc525916865)

[18 Conclusion 35](#_Toc525916866)

[19 References 36](#_Toc525916867)

# Introduction

As part of my studies in M2 Data & Knowledge, an internship must be carried out at the second semester to obtain professional experience and to apply the skills acquired during this year’s study.

Under this consideration, I chose to do my internship to implement the requirement of *Lounjee* to build a recommendation system which could give a user a list of recommendation to connect. And of cause this project allows me to acquire more knowledge in data mining, machine learning and deep learning to reinforce what I learned in this field.

In this report, I will introduce the Lab and the company which I worked. And then, I will detail the mission of this internship. Finally, I will summarize the internship and talk about what it brought me and the different that I met.

# Work Schedule

The total work time per week is 35 hours

|  |  |  |
| --- | --- | --- |
|  | Morning | Afternoon |
| Monday to Friday | 9H00 – 12H00 | 13H30 – 17H30 |
| Saturday and Sunday | Day-off | Day-off |

# Presentation of Laboratory & Lounjee

The computer science department, Laboratory of Informatics Paris Descartes (LIPADE), belongs to the Faculty of Mathematics and Computer Science. With its more than 60 members, LIPADE forms a vibrant and international research environment. It is carrying out theoretical and applied research in the areas of intelligent agents and multi-agent systems, machine learning, computer vision and image processing, networks, databases and data analytics. Its members are actively participating in European and French research projects, as well as in collaborations with the industry.



4.1 Photo of LIPADE

Lounjee is a business match-making mobile service that connects you to those professionals you should know but you do not know yet.

With its match-making algorithm, Lounjee can suggest you those professionals who you do not know yet, but you should know because of beneficial mutual opportunities. Lounjee surfaces these valuable and unexplored connections that will boost your professional career and business.



4.2 Work environment

# Presentation of Project

## Existing context

Professional networking is an activity made famous by services like LinkedIn and, as more than 70% of jobs are found through networking (US labor statistics), every professional is looking at improving at it. Beyond searching for a new job, expanding and maintaining a healthy and supportive network is key for business success.

Unfortunately, very few people are effective at professional networking, creating an opportunity for better products/services to them have a better network and achieve the outcomes they want.

Existing services like LinkedIn suggest people to connect but primarily people who are like you (i.e. based on shared connections/industry like LinkedIn or based on interests like Shapr).

Instead, Lounjee uses the following factors to match professional profiles: What people are looking for, what they can offer, Industry, and Location. Lounjee implements these factors into an early matching logic, which has been tested with real users.

The results from matching on these basic dimensions were encouraging (50% users returning each week, 80% users returning each month, over 50% of users favoring a profile i.e. saving it for later).

This validated the hypothesis that matching for professional network is not about similarity of profiles but rather about matching people who have a need (are looking for something) with people who can fill that need (can offer that thing)

Therefore, the traditional way of matching two users’ needs to be changed.

## Presentation of project

For now, Lounjee used a traditional algorithm which find the same elements between two users.

For example, there are userA and userB, and the Lounjee’s algorithm compare the value of the same feature, if the value is the same, the algorithm will count the number of the same value with this feature and multiply it by weight which pre-define before.

Expression: Score = Σ Wi\*Fi

5.2.1 Lounjee scoring algorithm

And then transform the score with a certain proportion to a rating star.

Referring to the current situation, this way to calculate the match score is stock, it only considers the same value, and the weights are defined by person which lack of the reliability. And Lounjee has already get the status of connection between each user, and users’ profiles. But they do not have a scientific model to recommend users.

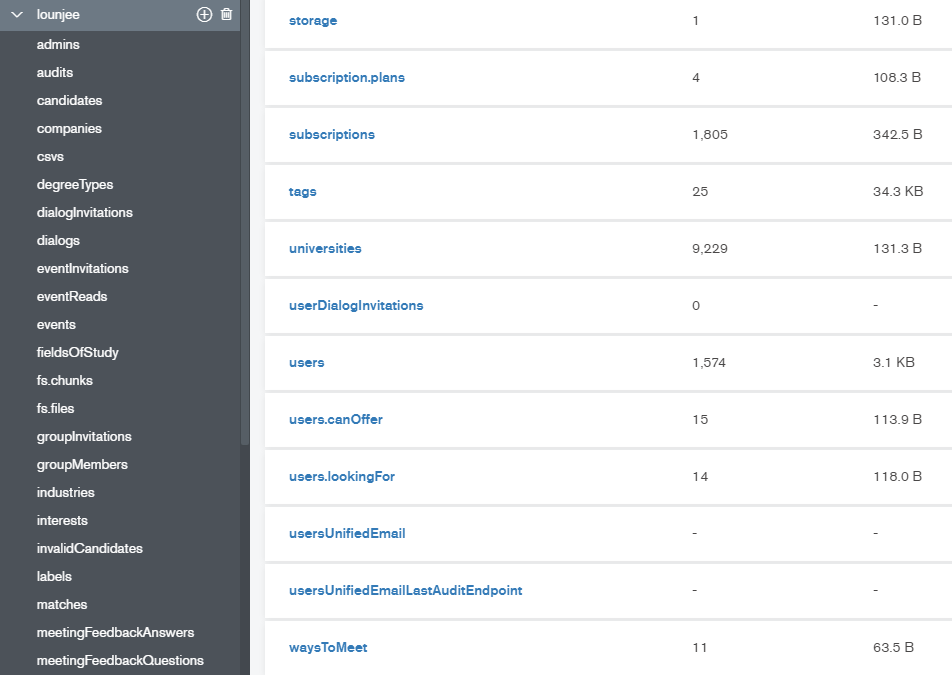
So, they proposed the project to predict which users should be recommended to the user.

# Project Implement: Data preprocessing

## Data Acquisition

Lounjee store the users’ data in database (MongoDB), and I get a dump which include all the real data and import them into my local MongoDB database. I am free to use these data on this project.

There are almost 40 collections in the database which include user’s profile, user match etc.



6.1.1 Lounjee data in MongoDB

And for now, there are 1,574 users who are registered in Lounee, also 29,114 connections status in database.

## Data Engineering

### Data Extraction

Python need install the package **PyMongo** to extract data from MongoDB, this is the most commonly used package to handle the data of MongoDB in python.

This project needs extract the data of collection Users and connection Matches to python.

### Data preprocessing

And then the package NumPy and Pandas are also used in the project. Pandas provides high-performance, easy-to-use data structures and data analysis tools. NumPy is the fundamental package for scientific computing with Python, and it contains:

* a powerful N-dimensional array object
* sophisticated (broadcasting) functions
* useful linear algebra, Fourier transform, and random number capabilities

With these two powerful packages, we will easy to analysis and treat the data which come from the MongoDB.

#### User preprocessing

There are almost 40 properties in user’s collection, but we don’t need all of them.



6.2.2.1.1 One user’s properties

In the case, I will use the properties as the follow:

* **Location**: User's recent login address
* **Experience**: Work experience (previous and current employment basically)
* **Education**: Education History
* **Group**: A way for multiple members to chat in same window
* **Skill**: Skills is identical in almost every way to Interests, except there is a different list of Skills
* **Industry**: The field of the user
* **Interest**: When people connect in the real world, it is often through a hobby/interest and later develops into a professional relationship.
* **Offer**: The user can offer something

And the result as the follow:



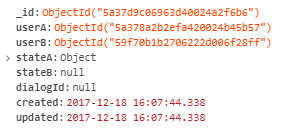
6.2.2.1.2 user’s feature with DataFrame

If there is no value, I keep mark it as None. And For the feature who has multi-value like vector. The separator is “,”. And I will flat it later in the data engineering part.

For the industry, I just record the id of the industry, and do not find the exact value of it. Multi-communication between database and python script will cost mush time, I have no raison to lost time on it.

#### User Match preprocessing

In the matches collection, there are userA’s id and userB’s id, also the state of their connection.



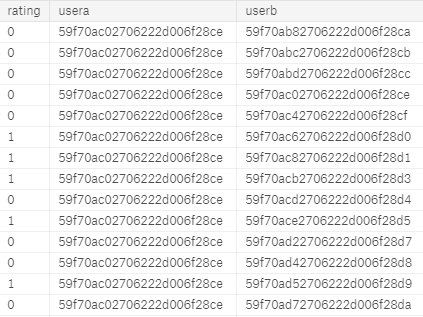
6.2.2.2.1 One connection between two users

States:

* Postponed
* Accepting
* Accepted
* Inviting
* Reporting

But so far, I only record the connection which state is [Postponed, Accepting, Accepted, Inviting]. If the connection’s state is one of above state, we will mark this connection as 1, that means the recommendation will recommend userB to userA. And then for rest connection, mark them as 0.

So finally, we will get a table as follow:



6.2.2.2.2 user match between two users

The usera is the user who ask to give him a recommendation list, and userb is the user who will recommend or not to usera, and if the rating is 0, that means do no recommend this userb to usera, if 1, recommend this userb to usera.

# Data Analysis

## Data integrity analysis

By reading the data, I find that some users have complete data, complete data means the user has full information in his profile. But there are some users’ data who are incomplete.

There are several reasons that users’ profiles have incomplete data:

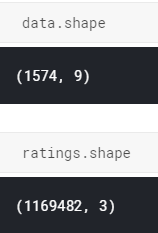
1. Some users’ data are imported from another database or file, there is no relative properties before, so the value is None for now.
2. The system does not ask user must fill the value of these properties.

Therefore, I need to find a way to solve it.

## Data balance analysis

Most real-world classification problems display some level of class imbalance, which is when each class does not make up an equal portion of your data-set. It is important to properly adjust your metrics and methods to adjust for your goals. If this is not done, you may end up optimizing for a meaningless metric in the context of your use case.

In the case, there are 1,574 users, so there are 1,169,482 user matches.



7.2.1 Numbers of user and user match

But there are only 29,122 user match’s value equal 1. That means the rest 1,140,360 user match equal 0. So, I conclude that this is an imbalanced data problem. And I should solve it.



7.2.2 Data distribution

So far, there are two solutions to solve the imbalanced data problem:

* Define the class (0 and 1) weight, give the bigger weight for the 1 whose quantity is very small.
* Sampling the data

### Classes Weight

Thanks to Sklearn who provide a way to calculate the class weight, we need to import the function from Sklearn to learn the weight:

*from sklearn.utils.class\_weight import compute\_class\_weight*

and then we can define the weight with

*class\_weight\_list = compute\_class\_weight('balanced', np.unique(y), y)*

*class\_weight = dict(zip(np.unique(y), class\_weight\_list))*

And later we can directly input the class\_weight into the machine learning model.

### Sampling

A widely adopted technique for dealing with highly unbalanced datasets is called resampling. It consists of removing samples from the majority class (under-sampling) and / or adding more examples from the minority class (over-sampling).

Despite the advantage of balancing classes, these techniques also have their weaknesses (there is no free lunch). The simplest implementation of over-sampling is to duplicate random records from the minority class, which can cause overfitting. In under-sampling, the simplest technique involves removing random records from the majority class, which can cause loss of information.

# Data engineering

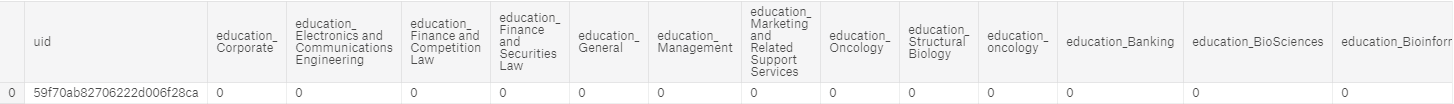
## One hot encoding

One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction.

As I mentioned in *User preprocessing* part, the user’s feature includes multi-value, or we can call the value is a list or vector for some features. These vectors are separated by “,”. For usual, we could directly use *pandas.get\_dummies* to categorical the values.

But for this case, we also need to consider the vactor, so I should give the parameter sep with ‘,’ for get\_dummies() function, and give a prefix with the original feature’s name to prevent duplicate values appear in different feature.

*df[name].str.get\_dummies(sep=',').add\_prefix(name+'\_')*



8.1.1 user’s feature after the one hot encoding

And the shape of the user is changed from (1574,8) to (1574,1494).

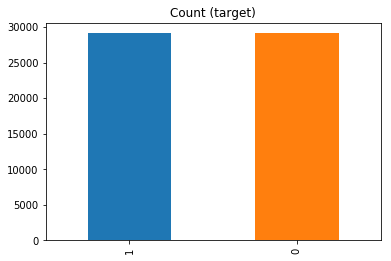
So, there are 1486 more dimensions for users.

## Random under sampling & random over sampling

Because this process will add or remove data from the dataset, that’s why I put this process after the one hot encoding. Because if we do this part first, maybe we will lose some users and their values. So, the result of one hot encoding will be different.

For the project, I give 2 methods for Lounjee to use, the random under sampling or the random over sampling.

But for me, I prefer the random under sampling. Because this will reduce the number of the user’s data. And, I think there exist lots of user match between whose state should be a question, the user does not accept it and do not refuse it. So, it shouldn’t be 1 or 0. These user matches are the noise data, so we should remove them to give our model more generalization ability.



8.2.2 After random under sampling



8.2.3 After random over sampling

And then, because the sampling adds more users or remove users from the dataset, so we need to reset the index of the dataset.

## Merge datasets

For now, I get two datasets, one for the user match and one for the users’ features.

So, I need merge them together to a one dataset which have the profile feature of usera and profile feature of userb also the 0 or 1 to judge recommend or not.

First for step, we should replace the usera’s id in the user match dataset with the feature of users whose id equal the usera’s id.

And then do the same operation for userb.

The advantage to replace id to feature:

* Id is unique, it only presents the one user, if the different user’s have the same profile, we can use one row to present them.
* With id, the model cannot learn the relationship between each user’s feature. The other elements have no chance to participle the training of model.

Finally, we will get the final dataset based on the user match dataset. And the shape of this dataset should be: (, 2271)

So far, the data engineering part is done, and after we get the final, we should collect the useless variables to clean the memory or computer.

# Model Introduction

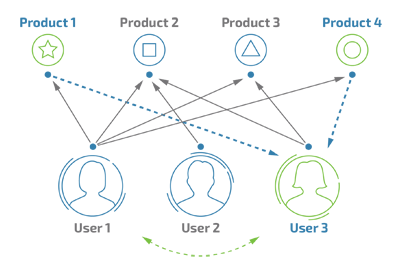
The supervised learning means that the training data has both features and tags. Through training, the model can find the relation between the features and the classes by itself.

Supervised learning has two types, regression and classification. According the data analysis and data engineering, we know that this is a binary classification problem.

## Popular Model: Collaborative filtering

Collaborative filtering is based on user experience and advice with similar attributes or interests as the basis for providing personalization. Collaborative filtering helps to collect users with similar preferences or attributes and provide their opinions to users in the same cluster as a reference to satisfy people's mentality of referencing others' opinions before making decisions.

The user-based collaborative filtering algorithm is to discover the user's likes of goods or content (such as product purchase, collection, content comment or sharing) through the user's historical behavior data, and measure and score these preferences. The relationship between users is calculated based on the attitudes and preferences of different users for the same product or content. Product recommendations are made between users with the same preferences. To put it simply, if both A and B users have purchased three books, x, y, and z, and given a 5-star rating. Then A and B belong to the same class of users. It is also possible to recommend the book w that A has seen to User B.



9.1.1 Collaborative filtering

But in this case, I cannot use this model, because for a start-up company, they cannot enough data to give the model to calculate the similar preferences or attributes between users.

And, there are the problem of cold-start, so there must have the problem that the user who are sign up with a total new profile which cannot find any similar preferences. So, the CF system will give a random result.

Therefore, this is an abandon model for me, and I will detail more about it in the capture failed version model.

## Traditional model: Matrix factorization model

Matrix factorization can be considered as a kind of information compression, let’s understand it as follow.

The users and content are not isolated, user preferences have similarities, and content has similarities. Compression is the process of quantifying users and content into k-dimensional vectors. Compressing the user vector dimension, making the vector dimension smaller, is itself a form of information compression. Various calculations can also be performed between vectors, such as cosine similarity, to quantify the distance between vectors, similarity, etc.

If there are m usersa and n usersb in the database, the size of the usera userb matrix is m×n. Each unit (i,j) uses Rij to indicate whether the usera has connected to the userb, ie 0 or 1. We represent the usera and the userb separately in a vector similar to Word2Vec, representing each usera i as a d-dimensional vector Xi and representing each userb j as a d-dimensional vector Yj . We are looking for Xi and Y j such that Xi × Yj and the usera userb matrix Rij are as close as possible, as shown. Thus, for a usera userb pair that has not appeared, the expression of Xi × Yj can be used to predict the score of any usera on the userb.

Usera

Userb

Y

X

R

9.2.1 Matrix Factorization

And we can use mathematical expressions to write this:

∑i,j (Rij − Xi × Yj )^2

## Advanced model: Deep learning model

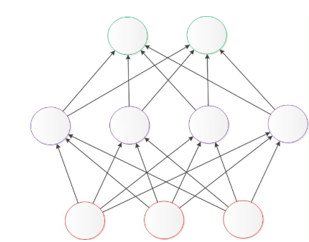
But here still has a more advanced model, deep learning.

We all know that Deep learning refers to a set of algorithms on a multi-layer neural network that uses various machine learning algorithms to solve various problems such as images and texts.

Deep learning can be classified into neural networks, but there are many changes in the specific implementation. The core of deep learning is feature learning, which aims to obtain hierarchical feature information through a layered network, thus solving the important problem that requires manual design features.

Deep learning is a framework that contains several important algorithms. For different problems (image, voice, text), different network models are needed to achieve better results.

And because the dataset what we finally get is a binary classification problem, so we will use the deep learning model to make the prediction and then I will find another machine learning model to compare their behavior.



9.3.1 Neaul Network

# Model Building

And now we can build the deep learning model to learn the data.

## Backend and framework

If you want to do something good, you must first sharpen it. There are many available choose now.

Caffe, PyTorch, TensorFlow, CNTK, Paddle, MXnet and others. But I will choose TensorFlow to be the backend of this project.

Because TensorFlow is a safe bet for most projects. No perfect but has huge community, wide usage. And pair with high-level wrapper (Keras, Sonnet, etc).



10.1.1 TensorFlow

And then, use Keras to build the Neural Network with our requirements.

Keras is a high-level neural network API, Keras is written in pure Python and is based on TensorFlow, Theano, and CNTK backends. Keras is born to support rapid experimentation and can quickly convert your idea into results.



10.1.2 Keras

## Deep learning model architecture

I need to adopt a flexible model, because the data of user include like: age, location, gender, skill etc. It must add these variables into model.

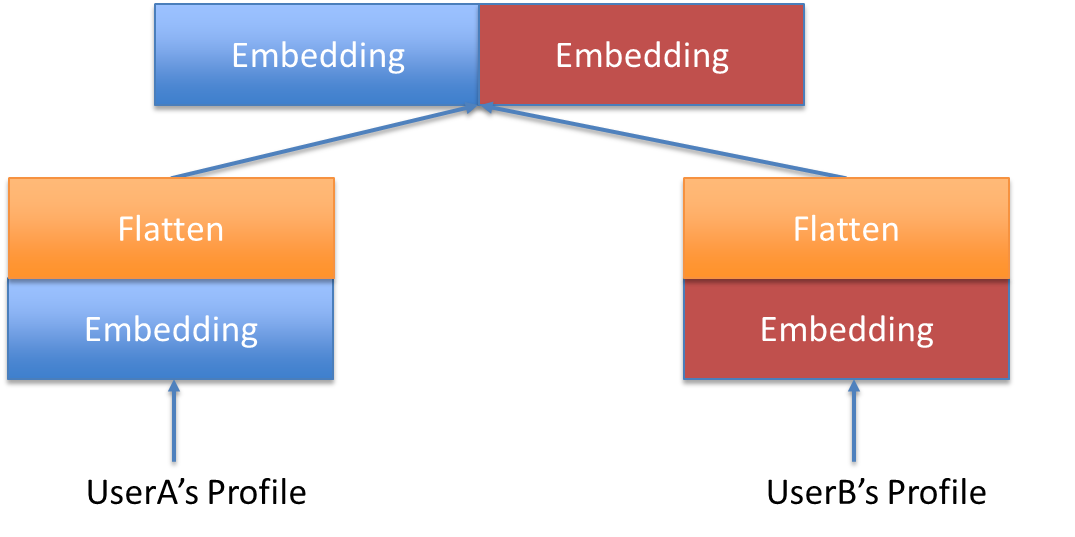
### Embedding layer

But this is not a traditional classification problem, there are two scopes (usera and userb) which include the same feature but different value. So, it cannot directly input them together.

So, first I need to put the usera and userb at different **Embedding** layer.

The raison to use embedding: Vectors encoded using the One-hot method will be very dimensional and sparse. So, embedding will help us to reduce the dimension. Its dimension reduction approach can be analogized to a fully connected layer (without an activation function), and the dimension is reduced by the weight matrix calculation of the embedding layer.

Finally, the input layer includes 2 embedding layers as follow:



10.2.1 embedding user’s profile

Note that Keras2.0 changed the API to merge two embeddings layers. New we need use keras.layers.merge.concatenate to merge the two Embedding to be one.

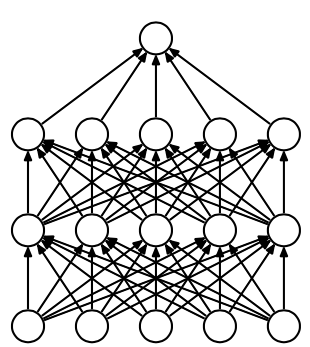
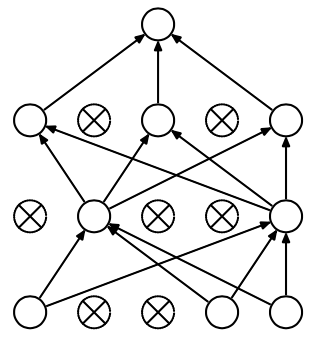
### Dropout layer

Dropout refers to the temporary discarding of neural network units from the network according to a certain probability during the training of the deep learning network.

Note that for the time being, for random gradient descent, each mini-batch is training a different network because it is randomly discarded.

So, dropout is a good way to prevent overfitting and improve performance of model.

For example, every time do a dropout, it's equivalent to finding a thinner network from the original network, as shown in the following figure:

10.2.2 Dropout principle

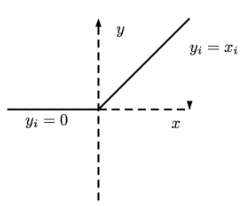
Therefore, for a neural network with N nodes, after dropout, it can be regarded as a collection of 2^n models, but the number of parameters to be trained at this time is unchanged.

So, to make our model more generalizable, I give each neuron the possibility of dropout.

### Activation: LeakReLU

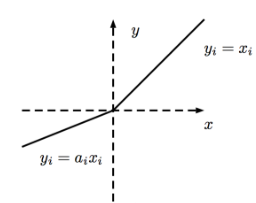
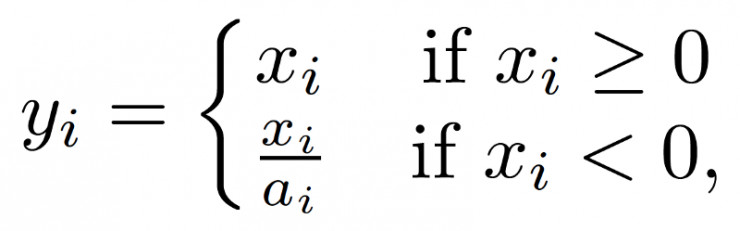
"Activation functions" can be divided into two categories - "saturation activation function" and "unsaturated activation function".

For usual, we use ReLU to be the activation function, this because ReLU is an unsaturated activation function, and it can solve the "gradient disappearance" problem. And, it speeds up convergence. The ReLU function sets all negative values to zero, and the remaining values are unchanged.



10.2.3.1 ReLU

But if we do not want to set all negative values to zero, we will use LeakReLU. LeakReLU assigns a non-zero slope to all negative values

10.2.3.2 LeakReLU

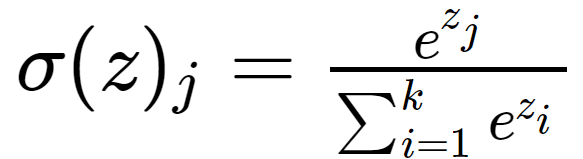
In this way, the data distribution is corrected, and some negative axis values are retained, so that the negative axis information is not completely lost.

### Last Activation: Softmax

Sigmoid activation is usually used for binary class problem.

Softmax activation is usually used for multi-class problem.

I decided to use Softmax for this project, even though there are only 1 and 0 classes. Because the recommendation system will not stop here. They implement a feedback function, the user will give a feedback for this match, and this feedback will be a rating star from 1 to 5. And then, Lounjee can use change the model to a multi-class model, each rating star could be a class. So, there will be 5 classes in the future.



10.2.4.1 Softmax

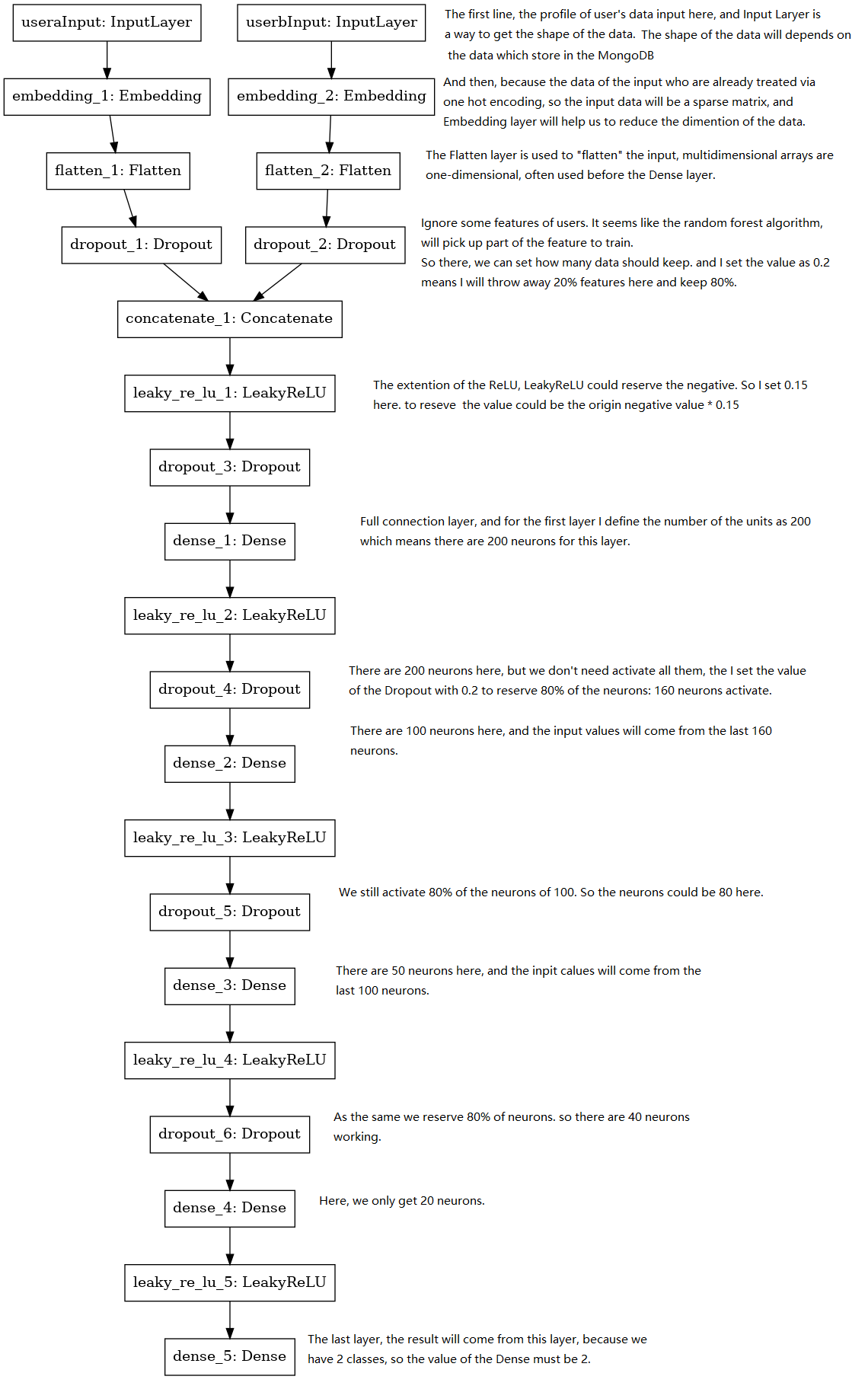
But because we think the 2 classes classification as a multi-classification problem, so, I need to category the value 0 and 1 to two class, it seems the one hot encoding what we used in the capture Data Engineering. But for the target, we should use sklearn.preprocessing.LabelEncoder to category them.

### Connected layer: Dense

Each node of the fully connected layer is connected to all nodes of the previous layer to combine the features extracted from the previous layer. Due to its fully connected nature, the parameters of the fully connected layer are also the most common.

For this project, I used 4 Dense layers to build the connected layer. And the initial units can override with the parameter k, and the second layer I reserve half units of the first layer. The third layer I reserve half units of the 2ed layer. And then reserve 40% units of third layer.

Finally, the last Dense reserve 2 units, one for class 1 and one for class 0. So, the complete structure is following:



10.2.5.1 Architecture of model

### Model Compile

Adam is a first-order optimization algorithm that can replace the traditional stochastic gradient descent process. It can iteratively update the neural network weights based on the training data.

And the loss function will use “categorical\_crossentropy”, this one usually uses on the multi-classes problem.

# Model training

## Callback Function

Usually, the neural network model will search for the optimal solution through many epochs, but we don't know when the model will reach the optimal solution, so set the callback to end the training early, when the loss score is no longer optimized.

## Batch size

I use the under sampling to balance data, so there are 58244 data, and we can define 1000 items / step

And based on the model, it needs almost 14 seconds per epoch with GPU speed up.

Epoch 1/200

58244/58244 [==============================] - 15s 255us/step - loss: 0.7120 - acc: 0.5101

Epoch 2/200

58244/58244 [==============================] - 14s 233us/step - loss: 0.5552 - acc: 0.7102

Epoch 3/200

58244/58244 [==============================] - 14s 234us/step - loss: 0.4588 - acc: 0.7844

Epoch 4/200

58244/58244 [==============================] - 14s 233us/step - loss: 0.4167 - acc: 0.8092

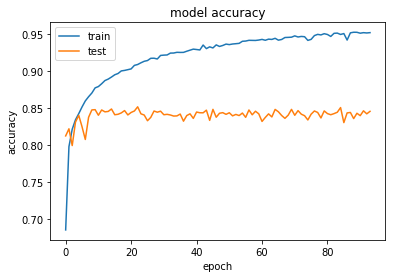
Epoch 5/200

58244/58244 [==============================] - 14s 233us/step - loss: 0.3849 - acc: 0.8275

## Model Performance Testing

The training is stop around 100 epochs, so, the callback is working at this moment.

During the training, I set the property validation\_split as 0.25, that means I will take 25% of data to be a test dataset, and 75% of data to be a train dataset, each epoch, the model will test the accuracy of train dataset and test dataset. But the validation\_split of Keras will take out the continuous data from the dataset, so we also need to set the shuffle as True, this means the model will shuffle the data before each epoch.



11.3.1 Model’s accuracy

The above result shows us that the model has a good behavior on the train data and the test data has almost 85% accuracy. So that means if the user whose profile as same as the train data, we can give him almost 95% current user to recommend, if not the same, he also can get almost 85% users.

# Prediction

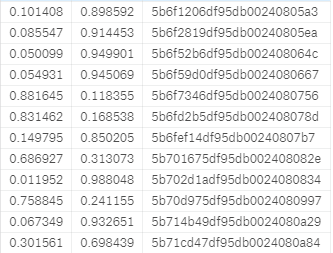
As we know, that Lounjee need to give a rating start for each recommend, and this will be used to order the recommendation list.

But our model only can predict one user match, so there is a function to generate a Dataframe which include usera (the user who we want to do the recommendation) and userb (all of the user without userb)

But our model is a binary classification model. So, I decide to do not predict directly the target. I just ask the model to predict the probability of the target, and that’s way I use Softmax to be the last activation.

And then, Lounjee could use the probability of class 1 to transform to the rating start according to a certain ratio

Finally, we can get a Dataframe like this:

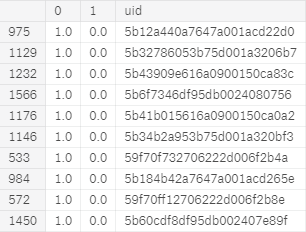
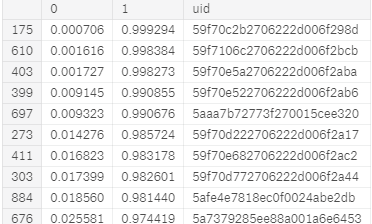


12.1 result of prediction

And the values of 0 and 1 are the probabilities of them.

# Result Sort

Because we need to sort the result according to the value of 1, so we can directly call the function of Pandas sort\_values to sort the value by 1, and then set the ascending as False to make sure the max value is on the top.



13.1 Top 10 and tail 10 users

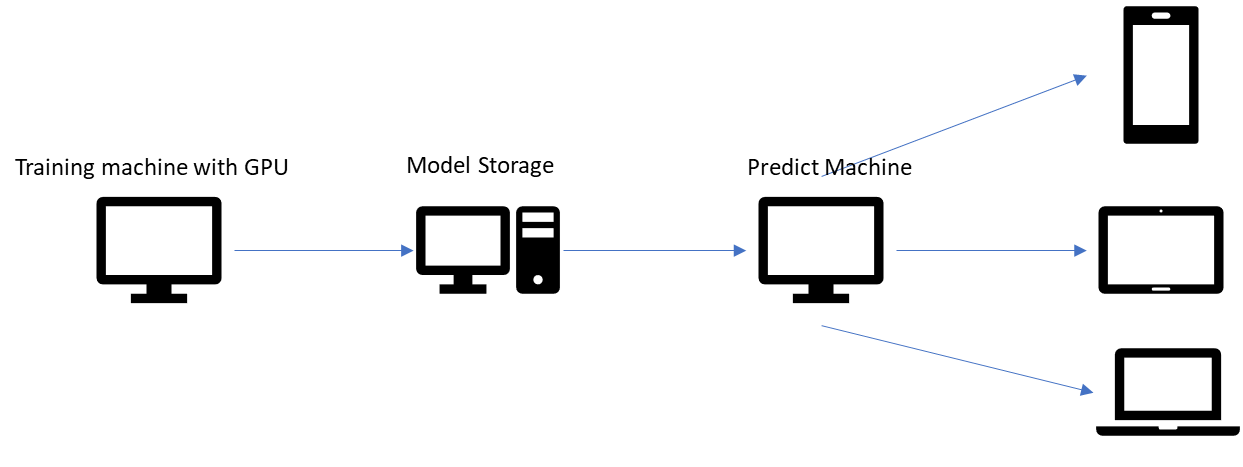
# model save & model load

But we all know that the deep learning will spend much time to learn the data, even if we set a callback “EarlyStopping” to stop the training when we get a good behavior. So, the user cannot wait for the training time. So, we can directly save the model to local, and then load it when we need it.

At the same times, we all know that GPU will spend much electricity, so if we train the model for once, and load it for a while. We can save much electricity for the world.

For example, we can prepare 3 machines, the first machine will focus on the training, the second machine will store the model which come from the first machine and then allow the third machine to load it and predict the require of users.

And if the new data come, or the new data will change the shape of the dataset, we can ask the first machine to re-train the data, and save the model to the 2em machine, and 2ed machine will load it for the next require coming. And the 2ed will not stop work will the training start.



14.1 Ideal architecture

# Restful service for Lounjee

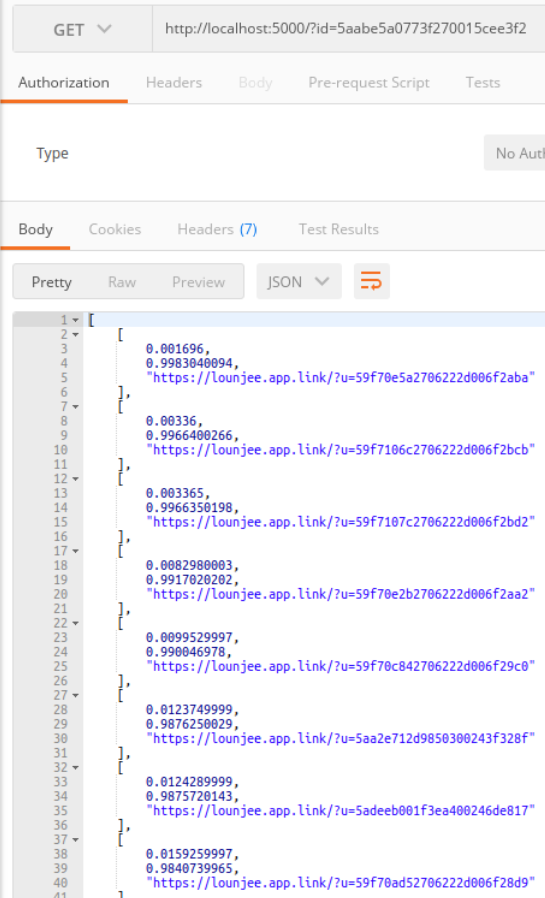
For now, we already have the list which include of recommendation information, but we must leave an API which will be used for service to call.

We can use the package **flask** to build a restful service on Python. The flask could launch on the predict machine of the ideal architecture, this restful service will load the model, and then predict the probability of target.

The API is in the root of site, and we can change the arg of id to set which id will do the recommendation. At the same time, It must set the Access-Control-Allow-Origin with \*, this means accept all the IP’ require.

For example, my id is “5aabe5a0773f270015cee3f2”, so the API to get my recommendation list is: <http://lounjee.com/?id=5aabe5a0773f270015cee3f2>

And then, I can get a json with the list like this:

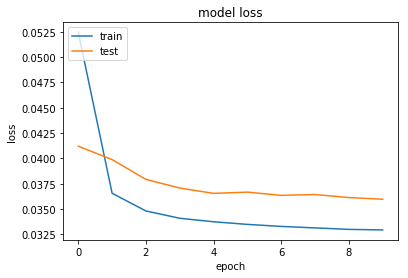


15.1 json response

# The versions what we did of recommendation system

## version 1: Deep Learning Regression

When I was building the deep learning model. I was lost in how to get the target to learn. So, I ask Lounjee gave me the score which product by their algorithm, this score is a discrete value. So I used the model which similar the model what I use now. But the last activation, I set it as linear and then the loss function is MSE (Mean square error)



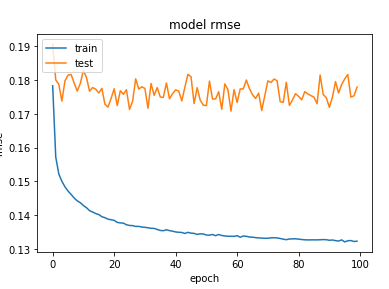
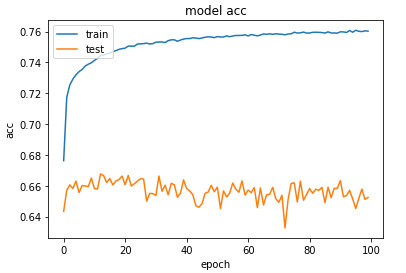
16.1 Regression result

## Version 2: Deep Learning Classification

We already see that the performance of the model to predict the score which product by Lounjee is good now. But we know that the score is product by Lounjee’s algorithm, which does not reflect the user's rating. So, Lounjee told me that they hold a feedback function which could get the rating of the user. And I think I can use this feedback to update the score which product by Lounjee, if Lounjee get enough feedback data. The model will learn the real valuable data.

So, I transform the score to the rating star according the weight rule which define by Lounjee. And then modify the last activation to be softmax. So now we can learn the multi-class problem.

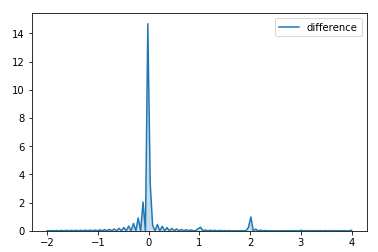
And there are 5 classes, 1,2,3,4,5.



16.2.1 Result

So we can find that alghout the reuslt of accuracy, but this is a rating star, so we can accept an appropriate deviation, so we should focus on the RMSE.

And I try to predict one user to another user’s score, and then I use the score to minus the score which product by Lounjee’s algorithm, and the result as follow:



16.2.2 The comparation

The result looks well, but we cannot use it, because the score is coming from the machine generate, not the real feedback of users.

That’s the raison I use the new model for now. The target is the real data from the user.

## Version 3: XGBClassifier （Split data）

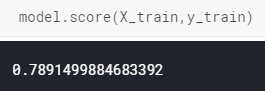
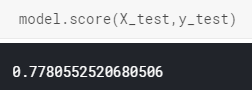
And I am also interested in the performance of the traditional machine learning model,

I try to run it on the xgbclassifier.

So first, I used the function of Sklearn to split the data with 4 parts.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)

And then training the model with the X\_train and y\_train data, and then test the X\_train and y\_train data again, the score for train data is 0.789 and I also test the test data, and also get 0.778.

16.3.1 xgbclassifier score

# New Data coming

So, for now, there is a problem that the data of Lounjee what they have now is incomplete. That means there are still lots of value which exist in database, but user do not use them as the profile data, so if the next user set his profile with a new value. The result of the one hot encoding will be changed. So, if the shape of the data changed, re-train the model is the only way to adapt it.

So, for this case, I gave Lounjee a solution is that create a super man whose profile include all the option of the feature. And then when we do the one hot encoding, it will flat all the value which exist in database.

And then, because the Keras support the streaming data. So, if a user connection come into the database, the listener should send a require to the model which should train the new data once time.

But of cause, the administer should set a cycle to train the model once a week or once a day. Because the project directly read the data from the database, so this is also a good way to adapt the new data.

# Conclusion

During this internship I studied so much, and I met also some difficulties.

The procedure of data acquisition is complicated and there are many a wide variety of data store in database, I need to overview all them to be sure that which data is useful for our project. This is not like a competition which already prepare the dataset for us.

And for beginning, I am lost in finding the label, the target. Because the Lounjee need the result is like the rating start with 1,2,3,4,5. But I cannot find the rating start in database, but finally I find a way to do with the probability.

But I also try to do other’s interesting test project, like use the multi-classification, regression etc. But because the problem of the data size, it cannot use now, But I think Lounjee will use them in the future.

# References

http://lipade.mi.parisdescartes.fr

https://api.mongodb.com/python/current

https://pandas.pydata.org

http://www.numpy.org

https://www.cnblogs.com/yeyinfu/p/7316692.html

<https://blog.csdn.net/qrlhl/article/details/60883604>

https://blog.csdn.net/dod\_jdi/article/details/78379781

https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45530.pdf